Exercises for Hoare Logic

Jean Pichon-Pharabod

2018/2019

This exercise sheet is based on previous exercise sheets by Kasper Svendsen and by Mike Gordon. Mike Gordon's exercise sheet also contains additional exercises: `https://www.cl.cam.ac.uk/teaching/1516/HLog+ModC/MJCG-HL-Exercises.pdf`.

Recommended exercises metatheory: 1, 22; practice: 2, 9, 35; specifications: 24, 25, 27; invariants: 12, 36, 37, 41; representation predicates: 47.

All the proof invariant exercises that do not involve separation logic can be formalised in Why3: `http://why3.lri.fr/try/`.

Exercise 1. Give a program C such that the following partial correctness triple holds, or argue why such a C cannot exist:

$$\{X = x \land Y = y \land x \neq y\}\ C\ \{x = y\}$$

Exercise 2. Show that the alternative assignment axiom

$$\overline{\{P\}\ X := E\ \{P[E/X]\}}$$

is unsound by providing P and E such that

$$\neg(\models \{P\}\ X := E\ \{P[E/X]\})$$

Exercise 3 (Soundness of Floyd's assignment axiom)**.** Show that the alternative assignment axiom

$$\frac{x \notin FV(P)}{\{P\}\ X := E\ \{\exists x.\, E[x/X] = X \land P[x/X]\}}$$

is sound.

Exercise 4 (Relative completeness of Floyd's assignment axiom)**.** Show that if we replace the assignment axiom by the following alternative assignment axiom

$$\frac{x \notin FV(P)}{\{P\}\ X := E\ \{\exists x.\, E[x/X] = X \wedge P[x/X]\}}$$

then the original assignment axiom is derivable.

Exercise 5. Show the soundness of the following rule:

$$\frac{\vdash \{P\}\ C\ \{Q\} \qquad \vdash \{P\}\ C\ \{R\}}{\vdash \{P\}\ C\ \{Q \wedge R\}}$$

Exercise 6. Show the soundness of the following rule:

$$\frac{\vdash \{P\}\ C\ \{R\} \qquad \vdash \{Q\}\ C\ \{R\}}{\vdash \{P \vee Q\}\ C\ \{R\}}$$

Exercise 7. Give a sound and relatively complete rule for a **repeat** C **until** B command (which is syntactic sugar for $C;$ **while not** B **do** C).

Exercise 8. Prove that the following backwards reasoning sequenced assignment rule is derivable from the normal proof rules of Hoare logic:

$$\frac{\{P\}\ C\ \{Q[E/X]\}}{\{P\}\ C; X := E\ \{Q\}}$$

Exercise 9. Prove or give a counterexample for the following triple:

$$\begin{array}{l}\{X = x \wedge Y = y\}\\ X := X + Y; Y := X - Y; X := X - Y\\ \{Y = x \wedge X = y\}\end{array}$$

Exercise 10. Give a proof outline, and in particular a loop invariant, for the following partial correctness triple:

$$\begin{array}{l}\{X = x \wedge Y = y \wedge Y \geq 0\}\\ \textbf{while}\ Y > 0\ \textbf{do}\ (X := X + 1; Y := Y - 1)\\ \{X = x + y\}\end{array}$$

Exercise 11. Give a variant to obtain a total correctness triple (you might need to strengthen the precondition and the invariant).

Exercise 12. Give a proof outline, and in particular a loop invariant, for the following partial correctness triple:

$$\begin{array}{l} \{X = x \wedge Y = y \wedge Y \geq 0\} \\ Z := 0; \\ A := 1; \\ \textbf{while } A \leq Y \textbf{ do } (Z := Z + X; A := A + 1) \\ \{Z = x \times y\} \end{array}$$

Exercise 13. Give a variant to obtain a total correctness triple (you might need to strengthen the precondition and the invariant).

Exercise 14. Recall that

$$\begin{array}{l} \vdash \forall x.\, gcd(x, x) = x \\ \vdash \forall x, y.\, gcd(x, y) = gcd(y, x) \\ \vdash \forall x, y.\, x > y \Rightarrow gcd(x, y) = gcd(x - y, y) \end{array}$$

Give a proof outline, and in particular a loop invariant, for the following partial correctness triple:

$$\begin{array}{l} \{X = x \wedge Y = y \wedge x > 0 \wedge y > 0\} \\ \textbf{while } X \neq Y \textbf{ do } (\textbf{if } X > Y \textbf{ then } X := X - Y \textbf{ else } Y := Y - X) \\ \{X = Y \wedge X = gcd(x, y)\} \end{array}$$

Exercise 15. Give a variant to obtain a total correctness triple (you might need to strengthen the precondition and the invariant).

Exercise 16. Give a proof outline, and in particular a loop invariant, for the following partial correctness triple:

$$\begin{array}{l} \{X = x \wedge Y = y\} \\ Z := 0; \\ \textbf{while not } (X = 0) \textbf{ do} \\ \quad \left(\begin{array}{l} (\textbf{if } X \textbf{ mod } 2 = 1 \textbf{ then } Z := Z + Y \textbf{ else skip}); \\ Y := Y \times 2; \\ X := X \textbf{ div } 2 \end{array} \right) \\ \{Z = x \times y\} \end{array}$$

Hint: $X = (X \textbf{ div } 2 + X \textbf{ div } 2 + X \textbf{ mod } 2)$.

Exercise 17. Give a variant to obtain a total correctness triple (you might need to strengthen the precondition and the invariant).

Exercise 18 (Fast exponentiation)**.** Give a proof outline, and in particular a loop invariant, for the following partial correctness triple:

$$
\begin{array}{l}
\{X = x \wedge N = n \wedge n \geq 0\} \\
Z := 1; \\
\textbf{while } N > 0 \textbf{ do} \\
\quad \left(\begin{array}{l}
(\textbf{if } N \textbf{ mod } 2 = 1 \textbf{ then } Z := Z \times X \textbf{ else skip}); \\
N := N \textbf{ div } 2; \\
X := X \times X
\end{array} \right) \\
\{Z = x^n\}
\end{array}
$$

Exercise 19. Give a variant to obtain a total correctness triple (you might need to strengthen the precondition and the invariant).

Exercise 20 (Turing's large routine)**.** Give a proof outline, and in particular loop invariants, for the following partial correctness triple:

$$
\begin{array}{l}
\{N = n \wedge n \geq 0\} \\
R := 0; \\
U := 1; \\
\textbf{while } R < N \textbf{ do} \\
\quad \left(\begin{array}{l}
S := 1; V := U; \\
\textbf{while } S \leq R \textbf{ do} \\
\quad (\ U := U + V; S := S + 1\ ); \\
R := R + 1;
\end{array} \right) \\
\{U = \mathit{fact}(n)\}
\end{array}
$$

Exercise 21. Give variants to obtain a total correctness triple for the same pre- and postcondition and command.

Exercise 22. Prove soundness of the separation logic heap assignment rule by proving that

$$\models \{E_1 \mapsto t\}\ [E_1] := E_2\ \{E_1 \mapsto E_2\}$$

Exercise 23. Formalise and prove that if $X \mapsto t_1 \wedge Y \mapsto t_2$, then X and Y alias, and t_1 and t_2 are equal.

Exercise 24. Give a triple specifying that a command C orders the values of X and Y, so that the smaller value ends in X, and the greater value in Y.

Exercise 25. Give a triple specifying that a command C computes into Z the sum of X and Y if R is 0, and their product otherwise.

Exercise 26. Give a triple specifying that a command C sorts a list starting at X.

Exercise 27. Give a triple specifying that a command C concatenates a list starting at X with itself.

Exercise 28. Give a triple specifying that a command C appends the value of V to the start of a list starting at X if R is 0, and to the end of a list at Y otherwise.

Exercise 29. Give a proof outline, and in particular a loop invariant, for the following partial correctness triple:

$$\begin{array}{l} \{N = n \land n \geq 0 \land X = 0 \land Y = 0\} \\ \textbf{while } X < N \textbf{ do } (X := X + 1; Y := Y + X) \\ \{Y = \sum_{i=1}^{n} i\} \end{array}$$

Exercise 30. Give a variant to obtain a total correctness triple (you might need to strengthen the precondition and the invariant).

Exercise 31 (Euclid's algorithm)**.** Give a proof outline, and in particular a loop invariant, for the following partial correctness triple:

$$\begin{array}{l} \{X = x \land Y = y\} \\ R := X; \\ Q := 0; \\ \textbf{while } Y \leq R \textbf{ do} \\ \quad (R := R - Y; Q := Q + 1) \\ \{x = R + y \times Q \land R < y\} \end{array}$$

Exercise 32. Give a variant to obtain a total correctness triple (you might need to strengthen the precondition and the invariant).

Exercise 33 (Divisibility by 13)**.** Give a proof outline, and in particular a loop invariant, for the following partial correctness triple:

$\{X = x \wedge X \geq 0\}$
$\textbf{while } X \geq 52 \textbf{ do}$
$\quad X := (X \textbf{ div } 10) + 4 \times (X \textbf{ mod } 10);$
$\textbf{if } X = 0 \textbf{ or } X = 13 \textbf{ or } X = 26 \textbf{ or } X = 39 \textbf{ then } Y := 1 \textbf{ else } Y := 0$
$\{Y = 1 \Leftrightarrow x \mod 13 = 0\}$

Exercise 34. Give a variant to obtain a total correctness triple (you might need to strengthen the precondition and the invariant).

Exercise 35. Give a proof outline for the following separation logic partial correctness triple:

$\{\mathit{list}(X, \alpha)\}$
$\textbf{if } X = \textbf{null then } Y := \textbf{null}$
$\textbf{else } (E := [X]; P := [X + 1]; Y := \textbf{alloc}(E, P); \textbf{dispose}(X); \textbf{dispose}(X + 1))$
$\{\mathit{list}(Y, \alpha)\}$

Exercise 36. Give a proof outline, and in particular a loop invariant, for the following separation logic partial correctness triple:

$\{\mathit{list}(X, \alpha)\}$
$Y := \textbf{null};$
$\textbf{while } X \neq \textbf{null do}$
$\quad (Z := [X + 1]; [X + 1] := Y; Y := X; X := Z)$
$\{\mathit{list}(Y, \mathit{rev}(\alpha))\}$

where *rev* is mathematical list reversal, so that

$$\begin{aligned} \mathit{rev}([]) &= [] \\ \mathit{rev}([h]) &= [h] \\ \mathit{rev}(\alpha +\!\!+ \beta) &= \mathit{rev}(\beta) +\!\!+ \mathit{rev}(\alpha) \end{aligned}$$

Exercise 37. Give a proof outline, and in particular a loop invariant, for the following separation logic partial correctness triple:

$\{\mathit{list}(X, \alpha)\}$
$N := 0;$
$Y := X;$
$\textbf{while } Y \neq \textbf{null do}$
$\quad (N := N + 1; Y := [Y + 1])$
$\{\mathit{list}(X, \alpha) \wedge N = \mathit{length}(\alpha)\}$

Exercise 38. Give a proof outline, and in particular a loop invariant, for the following separation logic partial correctness triple:

$$
\begin{array}{l}
\{N = n \wedge emp\} \\
\textbf{if } N \leq 0 \textbf{ then } X := \textbf{null} \\
\textbf{else} \left(\begin{array}{l}
X := \textbf{alloc}(0, \textbf{null}); \\
P := X; \\
I := 1; \\
\textbf{while } I < N \textbf{ do} \\
\quad (Q := \textbf{alloc}(I, \textbf{null}); [P+1] := Q; P := Q; I := I + 1)
\end{array}\right) \\
\{list(X, 0 :: \ldots :: n-1 :: []) \wedge N = n\}
\end{array}
$$

Exercise 39. Give a proof outline, and in particular a loop invariant, for the following separation logic partial correctness triple:

$$
\begin{array}{l}
\{list(X, \alpha)\} \\
Y := \textbf{alloc}(0, \textbf{null}); Y' := Y; \\
Z := \textbf{alloc}(0, \textbf{null}); Z' := Z; \\
\textbf{while } X \neq \textbf{null do} \\
\quad \left(\begin{array}{l}
[Y'+1] := X; Y' := X; X := [X+1]; \\
\textbf{if } X \neq \textbf{null then } ([Z'+1] := X; Z' := X; X := [X+1]) \textbf{ else skip}
\end{array}\right) \\
[Y'+1] := \textbf{null}; \\
[Z'+1] := \textbf{null}; \\
U := [Y+1]; \textbf{dispose}(Y); \textbf{dispose}(Y+1); Y := U; \\
U := [Z+1]; \textbf{dispose}(Z); \textbf{dispose}(Z+1); Y := U; \\
\{\exists \alpha_1, \alpha_2.\, length(\alpha) = length(\alpha_1) + length(\alpha_2) \wedge (list(Y, \alpha_1) * list(Z, \alpha_2))\}
\end{array}
$$

Exercise 40. Give a proof outline, and in particular a loop invariant, for the same separation logic partial correctness triple, but with the following postcondition:
$\{\exists \alpha_1, \alpha_2.\, shuffle(\alpha, \alpha_1, \alpha_2) \wedge (list(Y, \alpha_1) * list(Z, \alpha_2))\}$,
where

$$
\begin{aligned}
shuffle([], [], []) &\stackrel{def}{=} \top \\
shuffle(x :: \alpha, \beta, \gamma) &\stackrel{def}{=} (\exists \beta'.\, \beta = x :: \beta' \wedge shuffle(\alpha, \beta', \gamma)) \vee \\
&\qquad (\exists \gamma'.\, \gamma = x :: \gamma' \wedge shuffle(\alpha, \beta, \gamma'))
\end{aligned}
$$

Exercise 41. Give a proof outline, and in particular a loop invariant, for the following separation logic partial correctness triple:

$$
\begin{array}{l}
\{\mathit{list}(X,\alpha) \wedge \mathit{sorted}(\alpha) \wedge Y = y\} \\
\mathbf{if}\ X = \mathbf{null}\ \mathbf{then}\ X := \mathbf{alloc}(Y, \mathbf{null}) \\
\mathbf{else} \left(\begin{array}{l}
P := X; E := [P]; \\
\mathbf{if}\ Y \leq E\ \mathbf{then}\ X := \mathbf{alloc}(Y, X) \\
\mathbf{else} \left(\begin{array}{l}
Q := P; \\
\mathbf{while}\ E < Y\ \mathbf{and}\ P \neq \mathbf{null}\ \mathbf{do} \\
\quad (Q := P; P := [P+1]; E := [P]) \,; \\
R := \mathbf{alloc}(Y, P); \\
[Q+1] := R
\end{array}\right)
\end{array}\right) \\
\left\{\exists \alpha_1, \alpha_2.\ \begin{array}{l}
\alpha = \alpha_1 \mathbin{+\!\!+} \alpha_2 \wedge \\
(\forall i.\, 0 \leq i < \mathit{length}(\alpha_1) \Rightarrow \alpha_1[i] < y) \wedge \\
(\forall i.\, 0 \leq i < \mathit{length}(\alpha_2) \Rightarrow y \leq \alpha_2[i]) \wedge \\
\mathit{list}(X, \alpha_1 \mathbin{+\!\!+} [y] \mathbin{+\!\!+} \alpha_2)
\end{array}\right\}
\end{array}
$$

Exercise 42. Give a proof outline, and in particular a loop invariant, for the following separation logic partial correctness triple:

$$
\begin{array}{l}
\{\mathit{list}(X,\alpha)\} \\
\mathbf{if}\ X = \mathbf{null}\ \mathbf{then}\ Y := \mathbf{null} \\
\mathbf{else} \left(\begin{array}{l}
P := X; E := [P]; Y := \mathbf{alloc}(E, \mathbf{null}); Q := Y; P := [X+1]; \\
\mathbf{while}\ P \neq \mathbf{null}\ \mathbf{do} \\
\quad (E := [P]; Q_2 := \mathbf{alloc}(E, \mathbf{null}); [Q+1] := Q_2; Q := Q_2; P := [P+1])
\end{array}\right) \\
\{\mathit{list}(X,\alpha) * \mathit{list}(Y,\alpha)\}
\end{array}
$$

Exercise 43 (Index search)**.** Give a proof outline, and in particular a loop invariant, for the following separation logic partial correctness triple:

$$
\begin{array}{l}
\{X = x \wedge x \in_{list} \alpha \wedge list(Y, \alpha)\} \\
I := 0; Z := Y; S := 0; \\
\textbf{while } S = 0 \textbf{ do} \\
\quad \left(\begin{array}{l}
E := [Z]; \\
\textbf{if } E = X \textbf{ then} \\
\quad S := 1 \\
\textbf{else} \\
\quad (Z := [Z+1]; I := I + 1)
\end{array}\right) \\
\{\alpha[I] = x \wedge list(Y, \alpha)\}
\end{array}
$$

where $\in_{list}$ is list membership:

$$
\begin{aligned}
x \in_{list} [] &\stackrel{def}{=} \bot \\
x \in_{list} (y :: \beta) &\stackrel{def}{=} (x = y) \vee (x \in_{list} \beta)
\end{aligned}
$$

Exercise 44 (Prefix testing)**.** Give a proof outline, and in particular a loop invariant, for the following separation logic partial correctness triple:

$$
\begin{array}{l}
\{list(X, \alpha) * list(Y, \beta)\} \\
P := X; Q := Y; S := 1; \\
\textbf{while } S = 1 \textbf{ and } P \neq \textbf{null and } Q \neq \textbf{null do} \\
\quad \left(\begin{array}{l}
E := [P]; F := [Q]; \\
\textbf{if } E = F \textbf{ then} \\
\quad (P := [P+1]; Q := [Q+1]) \\
\textbf{else} \\
\quad S := 0
\end{array}\right) \\
\{list(X, \alpha) * list(Y, \beta) \wedge (S = 0 \Leftrightarrow \neg(\alpha \sqsubseteq \beta \vee \beta \sqsubseteq \alpha))\}
\end{array}
$$

where $\sqsubseteq$ is prefix relation:

$$
\begin{aligned}
[] \sqsubseteq \beta &\stackrel{def}{=} \top \\
h :: \alpha \sqsubseteq \beta &\stackrel{def}{=} \exists \gamma.\, \beta = h :: \gamma \wedge \alpha \sqsubseteq \gamma
\end{aligned}
$$

Exercise 45 (Substring testing)**.** Give a proof outline, and in particular a loop invariant, for the following separation logic partial correctness triple:

$$
\begin{array}{l}
\{\mathit{list}(X,\alpha) * \mathit{list}(Y,\beta)\} \\
S := 1; P := X; Q := Y; \\
\textbf{while } (S = 1 \textbf{ and } P \neq \textbf{null}) \textbf{ do} \\
\quad \left(\begin{array}{l}
\textbf{if } Q = \textbf{null then } S := 0 \\
\textbf{else} \\
\quad \left(\begin{array}{l}
E := [P]; F := [Q]; \\
\textbf{if } E = F \textbf{ then } P := [P+1] \\
\textbf{else skip}; \\
Q := [Q+1]
\end{array}\right)
\end{array}\right) \\
\{(S = 0 \Leftrightarrow (\alpha \sqsubseteq \beta)) \wedge (\mathit{list}(X,\alpha) * \mathit{list}(Y,\beta))\}
\end{array}
$$

where $\sqsubseteq$ is the (not-necessarily-contiguous) substring relation:

$$
\begin{aligned}
[] \sqsubseteq \beta &\overset{\mathit{def}}{=} \top \\
h :: \alpha \sqsubseteq \beta &\overset{\mathit{def}}{=} (\exists \gamma.\, \beta = h :: \gamma \wedge \alpha \sqsubseteq \gamma) \vee (\exists i, \gamma.\, \beta = i :: \gamma \wedge h :: \alpha \sqsubseteq \gamma)
\end{aligned}
$$

Exercise 46 (Bubble sort)**.** Give a proof outline, and in particular loop invariants, for the following separation logic partial correctness triple:

$$
\begin{array}{l}
\{\mathit{list}(X,\alpha)\} \\
D := 0; \\
\textbf{while } D = 0 \textbf{ do} \\
\quad \left(\begin{array}{l}
S := 1; P := X; \\
\textbf{while } P \neq \textbf{null do} \\
\quad \left(\begin{array}{l}
Q := [P+1]; \\
\textbf{if } Q \neq \textbf{null then} \\
\quad \left(\begin{array}{l}
E := [P]; F := [Q]; \\
\textbf{if } E \leq F \textbf{ then} \\
\quad P := Q \\
\textbf{else} \\
\quad (S := 0; [P] := F; [Q] := E)
\end{array}\right) \\
\textbf{else} \\
\quad \textbf{skip}
\end{array}\right); \\
\textbf{if } S = 1 \textbf{ then } D := 1 \textbf{ else skip}
\end{array}\right) \\
\{\exists \beta.\, \mathit{sorted}(\beta) \wedge \mathit{permutation}(\alpha,\beta) \wedge \mathit{list}(X,\beta)\}
\end{array}
$$

Exercise 47. Give a representation predicate $\mathit{btree}(t,\tau)$ for binary trees, given a mathematical representation $\tau ::= \mathit{Leaf} \mid \mathit{Node}\ n\ \tau_1\ \tau_2$, where n is an integer.

Exercise 48. Give a representation predicate $clist(t, \alpha)$ for circular lists.

Exercise 49. Give a representation predicate $list'(t, \alpha)$ for doubly-linked lists.

Exercise 50. Give a representation predicate $array(t, \alpha)$ for arrays starting at location t, the contents of which is represented by the mathematical list α.